PCT/AU99/01015

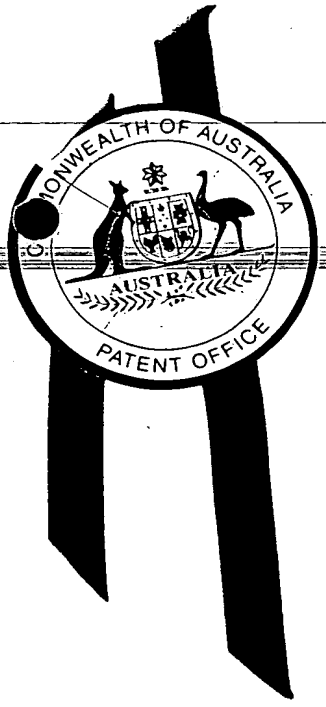
4

AU99/01015

09/831982

Patent Office
Canberra

I, KAY WARD, TEAM LEADER EXAMINATION SUPPORT AND SALES hereby certify that annexed is a true copy of the Provisional specification in connection with Application No. PP 7135 for a patent by TELSTRA R & D MANAGEMENT PTY. LTD. filed on 17 November 1998.



WITNESS my hand this
Twenty-first day of December 1999

KAY WARD
TEAM LEADER EXAMINATION
SUPPORT AND SALES

PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

Regulation 3.2

TELSTRA R&D MANAGEMENT
PTY. LTD.

A U S T R A L I A

Patents Act 1990

PROVISIONAL SPECIFICATION

for the invention entitled:

"A DATA ACCESS SYSTEM AND METHOD"

The invention is described in the following statement:

various display elements cannot be correctly dealt with, if at all, by the translation software of the voice browser. The various video and image formats, such as MPEG and JPEG, cannot be translated. Also a number of web sites utilise frames and tables in their HTML code which also gives rise to significant difficulties and complexity for the voice browser. The tendency for HTML pages is to include more and more visual elements, such as video and images, to attract users and this gives rise to significant difficulties for a voice browser.

A separate, independent, line of development is based on the work of the wireless application protocol (WAP) forum which is described at http://www.wapforum.org. The forum and WAP is concerned with the delivery of data on a wireless network and, in particular, the ability of wireless devices, such as mobile telephones, to access data resources, such as those available on the Internet. The WAP protocol is used by microbrowsers stored on wireless devices to access data information encoded in a particular format for the WAP protocol, such as the wireless markup language (WML). WML is a content language specifically developed for wireless devices with limited text displays and which use prompt driven navigation to display text and command menus. WML therefore allows delivery of text which can be selected and requested using displayed prompts and the keys of a wireless device. Details concerning WAP and WML are available at http://www.wapforum.org. A similar protocol and language for delivery of content to wireless devices has been developed by Unwired Planet, Inc. of California and details concerning their handheld device transport protocol (HDTP) and their handheld device markup language (HDML) are available at http://www.uplanet.com. HDML breaks text to be displayed on a wireless device into cards which can be moved between one another by selecting "accept" or "previous" keys on a handheld device. Soft keys are also available for inclusion in the cards to navigate between different cards based on display prompts. Data can also be keyed in on the wireless device in response to a received prompt and forwarded back to a HDTP server. Whilst the work of the WAP forum and Unwired Planet is a significant advance, access to the content still requires a person to have access to a wireless device, such as a mobile phone, which includes a microbrowser that can request and display content encoded in WML or HDML.

In accordance with the present invention there is provided a data access system,

The present invention also provides a voice browser stored on a computer readable storage medium, including:

code for converting voice data to voice for a caller connected to the browser;

code for processing a request for data from said caller sent on said call path;

code for sending said request to a location in a communications network determined by said request, said location including said data in a prompt navigation language; and

code for processing said data received from said location and translating said data in said prompt navigation language into said voice data.

A preferred embodiment of the present invention is hereinafter described, by way of example only, with reference to the accompanying drawings, wherein:

Figure 1 is a block diagram of a preferred embodiment of a data access system with a voice browser;

Figure 2 is a flow diagram of steps executed by a call flow module of the system; and

Figure 3 is a flow diagram of steps executed by a page translator of the system.

A data access system, as shown in Figure 1, includes an interactive voice response (IVR) platform 2 having a call flow module 8, a page translator module 10 and a text to speech module 16 stored on the platform. The IVR 2 also includes a telecommunications interface 18 for terminating calls received from a caller using a voice terminal 4, such as a standard telephone, so as to complete and connect to a communications path 20 between the caller 4 and the IVR 2. The IVR 2 also has an ISDN interface 22 to provide a permanent ISDN primary rate connection 24 to the Internet 12. The IVR 2 is a standard IVR platform which includes the interfaces 18 and 22 and the text to speech module 16. The IVR may, for example, be the First Contact IVR produced by Scitec. The text to speech module 16 may be the Learnout and Hauspie TruVoice module. The Internet 12 includes at least one server 14 which is able to communicate with the page translator 10 using a standard protocol, such as HTTP, and stores prompt navigation language data, such as WML or HDML data. The modules 8, 10 and 16 and the interfaces 18 and 22 provide a voice browser for use by a caller 4.

50 which is returned in a get link message, at step 60, to the translator 10. The translator 10 responds to the get link message by returning a URL to the call flow module 8 which is received at step 62. The URL returned by the translator 10 is checked at step 64 to determine whether a null URL has been returned, in which case the existing page continues to be announced to the caller 4, as the call flow module proceeds to step 58. If the URL is not a null, then it is stored by the call flow module 8 at step 66 and operation returns to step 36 to download the page corresponding to the new URL.

If the element el_num does not include a digit prompt, operation proceeds to step 52 to determine whether the element has a string prompt. A string prompt is one which calls for data entry by the caller so if the prompt is included in the element operation proceeds to step 54, where announcement to the caller is halted to await a sequence of key presses to be entered. The prompt will ask the user to enter the data used in the keys of the terminal and then press a final digit, such as "#". When the call flow module 8 detects that a sequence of key presses have been fully entered at step 56, the data inputted is passed to step 60 for formulation in a get link message for the translator 10.

If the element el_num does not include any prompts, as determined at steps 46 and 52, the variable el_num is incremented at step 58 and operation returns to step 40. The call flow module 8 will cease execution of the steps when the caller 4 disconnects from the call path 20.

The page translator module 10 operates in response to the messages received from the call flow module 8 and executes the steps described below with reference to Figure 3. The translator 10 accordingly waits for a message from the call flow 8 at step 82. On receiving a message, the message is stored at step 84 and action taken based on the form of the message is determined in steps 86, 90, 96 and 102.

If the message is an initialising message, as determined at step 86, operation proceeds to step 88, otherwise operation proceeds to step 90. At step 88, the translator 10 simply forwards a default home page URL to call flow 8 and operation returns to step 82.

determines the link designated in the get link message which includes data corresponding to DTMF signals returned by the caller 4. The get link message also includes the element number el_num for which the data has been returned from the caller. Using the element number, the translator is able to determine whether the return data corresponds to a link or a prompt. For a link, using the example described above, if a DTMF signal corresponding to 1 is returned then the URL for CNN is obtained from the element stored by the translator 10. If a 2 is returned, then the translator simply provides a null URL for the link. If the get link message and the element number corresponds to a prompt, then variable substitution may need to be made for the URL obtained from the corresponding stored element. Using the example described above, if the caller enters DBW# then the initials DBW are substituted in the variable, and the link URL is determined to be http://foo/cgi?DBW. The link URL determined at step 104 is then returned at step 106 to the call flow module 8, and operation returns to step 82.

As will be apparent from the above, the call flow module 8 and page translator 10 provide an efficient voice browser which has a number of significant advantages. As WML and HDML are languages which are used by wireless device microbrowsers, content developers are able to develop content for these microbrowsers as well as for people with telephones who can dial the IVR platform 2. This content is therefore available to people who would ultimately not have had access to the content, because they do not possess a device with a microbrowser or a computer system with access to the Internet.

The syntactical complexity and visual display elements of HTML ensure that HTML pages cannot be unambiguously or correctly decoded by a voice browser. Whilst HTML content can be written to take a voice browser into account, this is invariably not the case, whereas the content and flow of control are clear and well defined in prompt navigation content languages such as WML and HDML. The prompt navigation languages are efficiently utilised by the IVR 2. The translation of WML or HDML pages by the IVR 2 also does not introduce a significant processor load, and accordingly a larger number of simultaneous connections to callers 4 can be maintained without complex or expensive hardware.

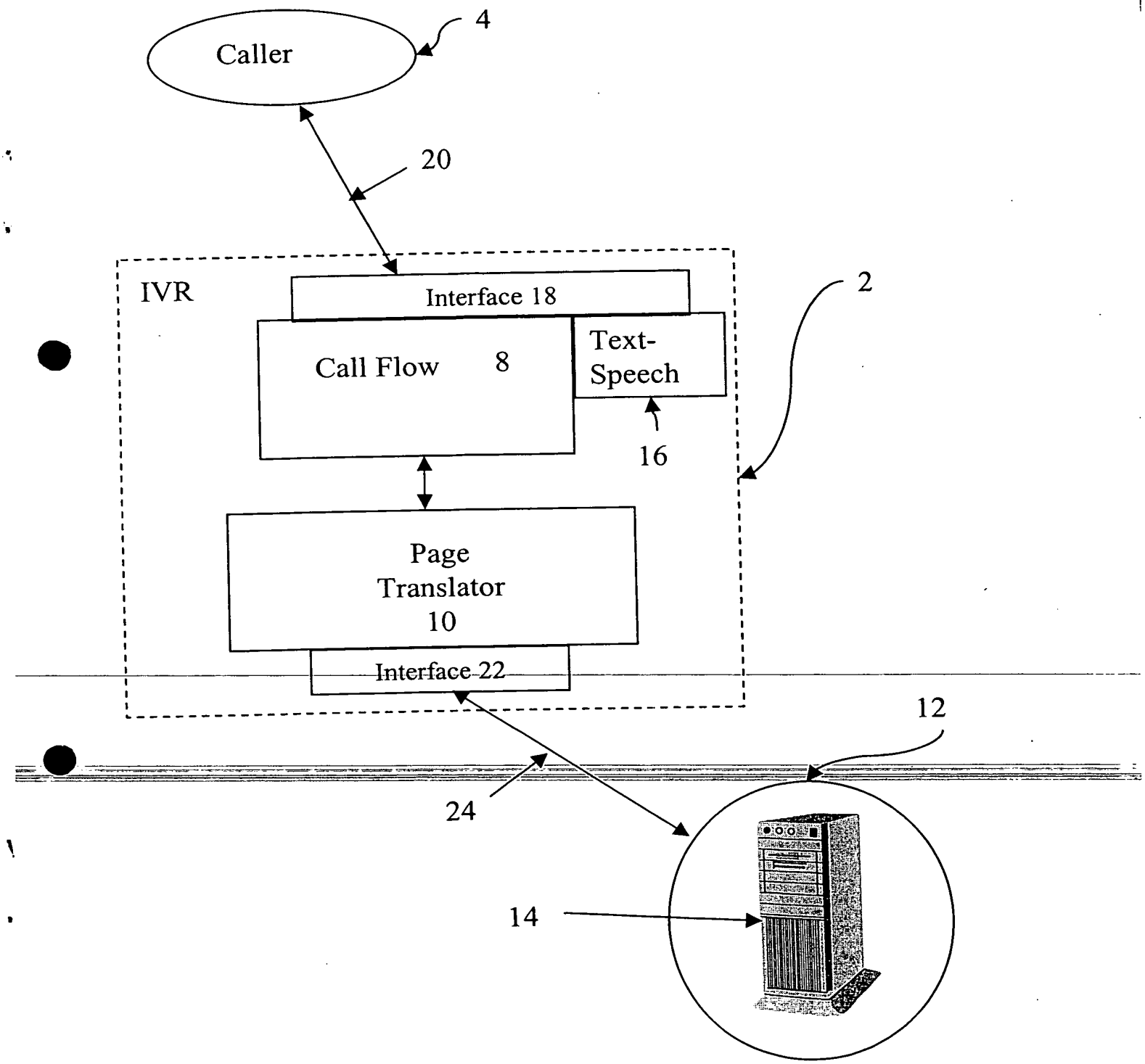

Caller
4
20
IVR
Interface 18
Call Flow 8
Text-
Speech
16
2
Page
Translator
10
Interface 22
12
24
14

Figure 1

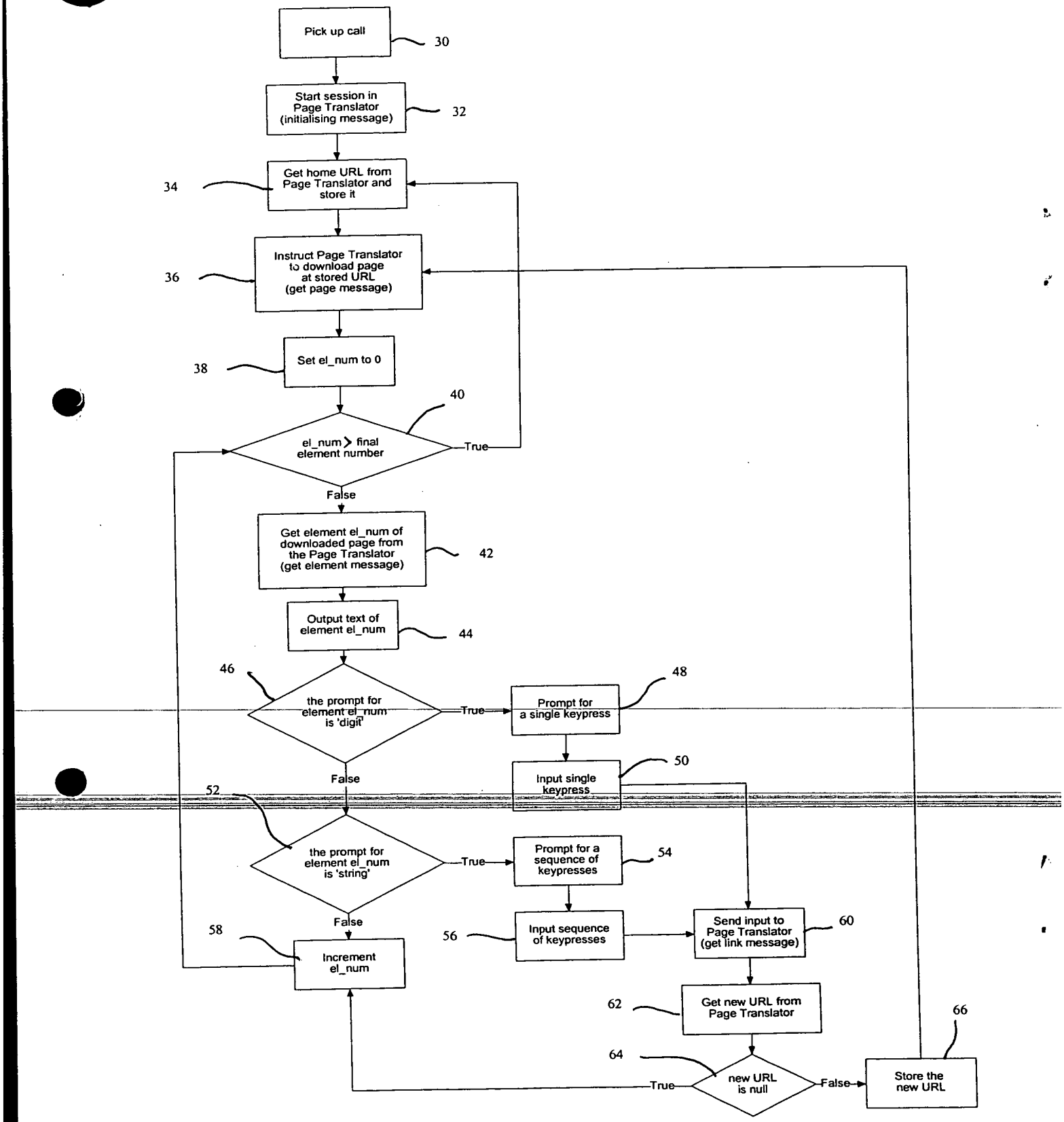

Pick up call
30
Start session in Page Translator (initialising message)
32
Get home URL from Page Translator and store it
34
Instruct Page Translator to download page at stored URL (get page message)
36
Set el_num to 0
38
40
el_num > final element number
True
False
Get element el_num of downloaded page from the Page Translator (get element message)
42
Output text of element el_num
44
46
the prompt for element el_num is 'digit'
True
Prompt for a single keypress
48
Input single keypress
50
False
52
the prompt for element el_num is 'string'
True
Prompt for a sequence of keypresses
54
False
58
Increment el_num
56
Input sequence of keypresses
Send input to Page Translator (get link message)
60
62
Get new URL from Page Translator
64
new URL is null
True
False
Store the new URL
66

Figure 2

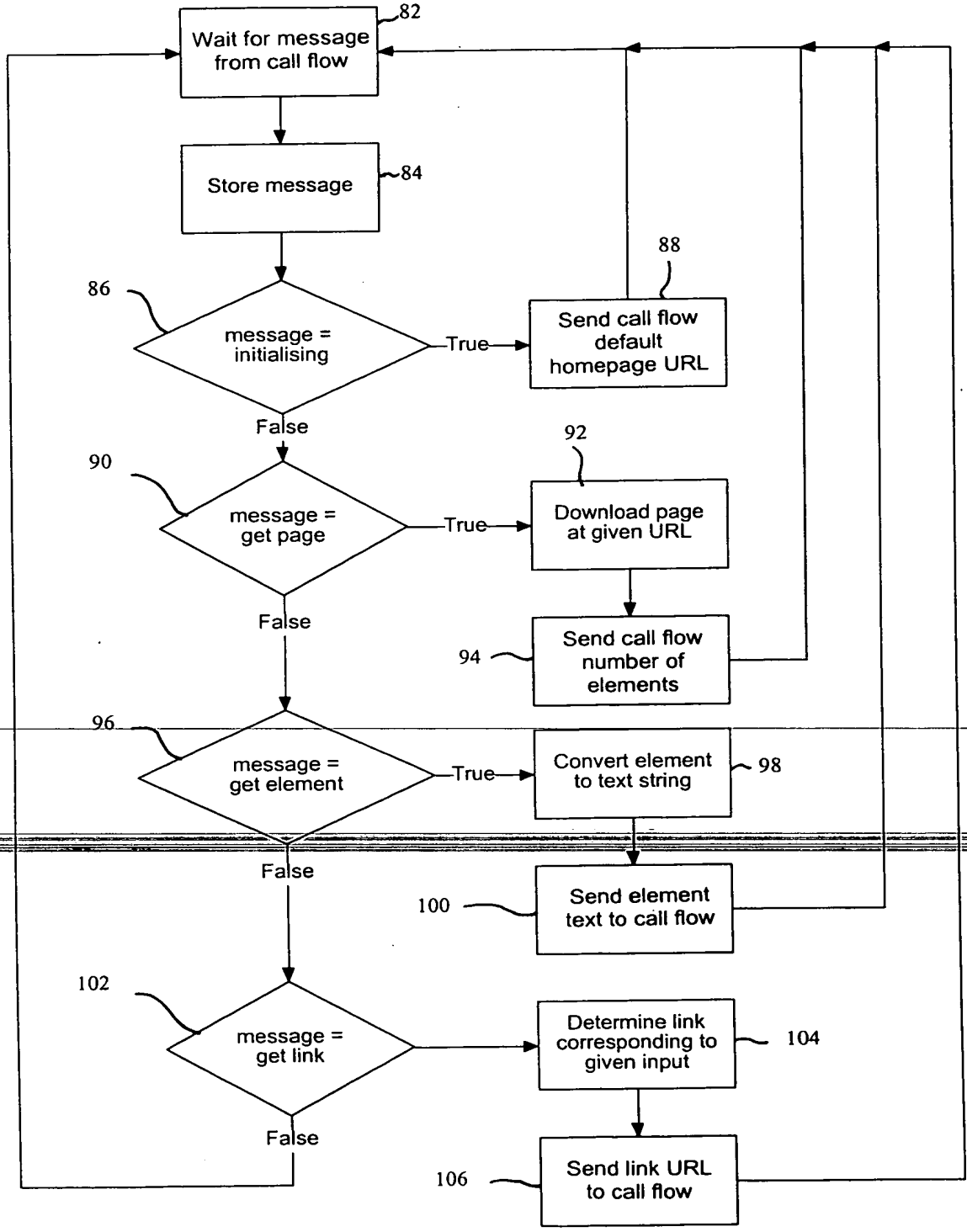

Wait for message from call flow
82
Store message
84
message = initialising
86
True
Send call flow default homepage URL
88
False
message = get page
90
True
Download page at given URL
92
False
Send call flow number of elements
94
message = get element
96
True
Convert element to text string
98
False
Send element text to call flow
100
message = get link
102
Determine link corresponding to given input
104
False
Send link URL to call flow
106

Figure 3